S32G MCAL UART DMA Configuration

by John Li (nxa08200)

本文说明如何配置MCAL UART模块为DMA模式。

默认的MCAL UART模块是使用的PIO模式。

本文采用软件版本为MCAL RTD 4.0.2。

历史	说明	作者
V1	● 创建本文	John.Li

目录

1 背景与资料说明

1.1 背景说明

M7 Core 使用的通用串行通讯接口，除了 CAN 之外，一般有 I2C，SPI，与 UART 接口。而对 SPI 及 UART 往往有高速 DMA 的要求。目前 MCAL 的 SPI 驱动默认配置为 Master/DMA 模式，而 MCAL 的 UART 模块驱动示例为 PIO 模式，不过 RTD_UART_IM/UM 文档说明了如何配置为 DMA 模式，本文也说明了配置方法。

1.2 所需资料说明

软件/工具	名称	如何获得
MCAL RTD	RTD_4.0.2 及相关文档：RTD_UART_IM.pdf，RTD_UART_UM.pdf	从www.nxp.com/s32g 个人帐号下载
S32G3 Refercence Manual	《S32G3RM》，包括其附件：《S32G3_DMAMUX_map.xlsx》	从www.nxp.com/s32g 下载

2 创建 UART 工程

2.1 打开工程

打开 EB tresos Studio 27.1 后 File->Import->General->Existing Projects into Workspace->Select root directiory Browse-> C:\NXP\SW32G_RTD_4.4_4.0.2\eclipse\plugins\Uart_TS_T40D11M40I2R0\examples\EBT\S32G3\Uart_Example_S32G399A_M7。

选中：Copy projects into workspace。

点击 Finish，导入工程。

在 Uart_Example_S32G399A_M7 上右击->Properties->Configuration Project->Code Generator->Default Generation Patch=..\..\generate。

所以工程文件是在：C:\EB\tresos\workspace\Uart_Example_S32G399A_M7。

在 Uart_Example_S32G399A_M7 上右击->Generate Project 后，生成的配置文件在：

C:\EB\tresos\generate。将此目录拷贝到 C:\NXP\SW32G_RTD_4.4_4.0.2\eclipse\plugins\Uart_TS_T40D11M40I2R0\examples\EBT\S32G3\Uart_Example_S32G399A_M7\generate 下。

2.2 修改波特率

Uart_Example_S32G399A_M7->someId(…)->Uart(…)->Uart->UartChannel->UartChannel_1->General：

- DesireBaudrate = LINFLEXD_UART_BAUDRATE_115200
- Uart Stop Bit Number = LINFLEXD_UART_IP_ONE_STOP_BIT

然后重新生成文件并拷贝。

2.3 编译与运行

参考文件：C:\NXP\SW32G_RTD_4.4_4.0.2\eclipse\plugins\Uart_TS_T40D11M40I2R0\examples\EBT\S32G3\Uart_Example_S32G399A_M7 \description.txt，来修改编译配置文件：C:\NXP\SW32G_RTD_4.4_4.0.2\eclipse\plugins\Uart_TS_T40D11M40I2R0\examples\EBT\S32G3\Uart_Example_S32G399A_M7 \project_parameters.mk

```
GCC_DIR = C:/NXP/S32DS.3.5/S32DS/build_tools/gcc_v9.2/gcc-9.2-arm32-eabi
TRESOS_DIR = C:/EB/tresos
T32_DIR = C:/T32
```

并在 cygwin 中编译：

```
make build
…
Compiling ../../../../..//Platform_TS_T40D11M40I2R0/startup/src/m7/gcc/startup_cm7.s
Linking main.elf
```

将S32G3 RDB3板设置为下载模式，连接上电源和Lauterbach，连接串口从PC到J1 UART1，PC串口终端配置为115200，8bits，0 parity bit，1 stop bits，运行脚本：C:\NXP\SW32G_RTD_4.4_4.0.2\eclipse\plugins\Uart_TS_T40D11M40I2R0\examples\EBT\S32G3\Uart_Example_S32G399A_M7\debug\run.cmm，打印出：

```
This example is an simple echo using UART
The board will greet you if you send 'Hello'
Now you can begin typing:
(此处输入Hello\r，需要修改代码fix bug，见下节)
Hello World
```

2.4 默认工程说明

Uart_Example_S32G399A_M7->someId(…)->Uart(…)->Uart->UartChannel->UartChannel_1->General：

- Uart hardware channel = LinflexD_1 //S32G 支持 3 个串口，与 FlexLin 复用，注意一般 MCAL 测试时使用的是 UART1，因为 UART0 是 ROM Code 再使用，所以特别在下载模式下，ROM Code 一样会访问 UART0，可能导致测试结果与代码不符。
- UartClockRef = /Mcu/Mcu/McuModuleConfiguration/McuClockSettingConfig_0/UART_CLK
- DesireBaudrate = LINFLEXD_UART_BAUDRATE_9600 //一般会修改为通常使用的 LINFLEXD_UART_BAUDRATE_115200
- Uart Asynchronous Method= LINFLEXD_UART_IP_USING_INTERRUPTS //默认不使用 LINFLEXD_UART_IP_USING_DMA
- Uart TX/RX DMA Channel 没有配置，因为前项没有打开 DMA
- Uart Stop Bit Number = LINFLEXD_UART_IP_TWO_STOP_BIT//一般会修改为通常使用的 LINFLEXD_UART_IP_ONE_STOP_BIT
- Uart Word Length=LINFLEXD_UART_IP_8_BITS

Uart_Example_S32G399A_M7->someId(…)->Mcu(…)->Mcu->McuClockSettingConfig->McuClockSettingConfig_0->

McuCgm0ClockMux8：

- CGM2 Clock Mux8 Source= FIRC_CLK
- Clock Mux8 Frequency (LIN_BAUD_CLK) (dynamic range)= 4.8E7

McuClockReferencePoint-> UART_CLK：

- Mcu Clock Reference Point Frequency (0 -> 5000000000)= 4.8E7
- Mcu Clock Frequency Select=LIN_BAUD_CLK

主测试函数：主测试函数逻辑为

1. 用异步方式发送：#define WELCOME_MSG "This example is an simple echo using UART\r\n\
   The board will greet you if you send 'Hello Board'\r\
   \nNow you can begin typing:\r\n" 并死循环等待发送成功。
2. While(1)中，用异步方式接收数据，并死循环等待接收到期待数据：#define EXPECT_RX_MSG "Hello\n"。
3. 比较发送和接收数据，如果相同，则以异步方式发送：Hellow world。

注意，因为#define EXPECT_RX_MSG "Hello\n"中是用的换行符”\n”，而大多数串口工具会使用回车键”\r”来发送，所以代码修改为：

```
#define EXPECT_RX_MSG "Hello\r"
```

来 fix 掉此 Bug。

3 配置 UART DMA 模式

3.1 参考资料

参考文档：《RTD_UART_UM.pdf》的：3.6.2 节: How to configure DMA。和文档《RTD_UART_IM.pdf》的：5.7 节 Data Cache Restrictions 和 6.3 节的 Calls to Notification Functions, Callbacks, Callouts。来配置 DMA 功能。

3.2 增加并配置 MCL 模块

在 Uart_Example_S32G399A_M 上右击，选择 Module Configurations，然后点中 Mcl(V4.0.2,AS4.4.0)，然后点击右向箭头加入工程。

Uart_Example_S32G399A_M7->someId(…)->Mcl(…)->Mcl->

General:

- Config Variant= VariantPreCompile
- Enable DMA Support =checked

Dma Logic Instance：点击+号增加 dmaLogicInstance_ConfigType_0

Dma Logic Channel：点击+号增加 dmaLogicChannel_Type_0 修改名字为：

- Interrupt Callback* =Linflexd_1_Uart_Ip_DmaTxCompleteCallback //参考源文件：Uart_ts_xxx/linflexd_Uart_Ip_Irq.h，及文档《RTD_UART_IM.pdf》：

Table 6.1 UART DMA Notification on S32G platform

Physical Unit	UART TX DMA Notification Name	UART RX DMA Notification Name
Uart_0	Linflexd_0_Uart_Ip_DmaTxComplete↩ Callback	Linflexd_0_Uart_Ip_DmaRxComplete↩ Callback
Uart_1	Linflexd_1_Uart_Ip_DmaTxComplete↩ Callback	Linflexd_1_Uart_Ip_DmaRxComplete↩ Callback
Uart_2	Linflexd_2_Uart_Ip_DmaTxComplete↩ Callback	Linflexd_2_Uart_Ip_DmaRxComplete↩ Callback

- Error Interrupt Callback=Linflexd_1_Uart_Ip_DmaTxCompleteCallback
- Enable Global Config=checked
- Enable Transfer Config=checked

->Global：

- Enable DMAMUX Source=checked
- DMAMUX Source*=DMA_IP_REQ_MUX0_LINFLEXD1_TX
- Enable DMA Request =checked
- Enable Error Interrupt=checked

->Transfer ： 使用默认值。

相同方法增加一个 RX Channel。不同点有：

- Enable Transfer Config*=uncheck。
- Level Priority = DMA_IP_LEVEL_PRIO2 // TX=DMA_IP_LEVEL_PRIO0，根据文档《RTD_SPI_UM》说明：

```
• If DMA uses fixed priority arbitration, then the priority must be Uart RX DMA Channel > Uart TX
DMA Channel.
```

具体请参考工程配置文件：
C:\EB\tresos\workspace\Uart_Example_S32G399A_M7\config\Mcl.xdm

然后修改源文件：
C:\NXP\SW32G_RTD_4.4_4.0.2\eclipse\plugins\Uart_TS_T40D11M40I2R0\examples\EBT\S32G3\Uart_Example_S32G399A_M7\src\main.c

```
#include "Mcl.h"
…
    /* Initialize MCL Driver */
  Mcl_Init(NULL_PTR);
```

最后修改文档：
C:\NXP\SW32G_RTD_4.4_4.0.2\eclipse\plugins\Uart_TS_T40D11M40I2R0\examples\EBT\S32G3\Uart_Example_S32G399A_M7\project_parameters.mk

```
MCAL_MODULE_LIST := BaseNXP Det Platform Mcu Port Resource EcuC Rte Mcl Uart
```

将 Mcl 添加进去。

3.3 修改 UART 模块

Uart_Example_S32G399A_M7->someId(…)->Uart(…)->Uart->

General：

- Uart DMA Enable=checked
- Uart Callback Capability=checked

UartChannel-> UartChannel_1：

- Uart Asynchronous Method*= LINFLEXD_UART_IP_USING_DMA

- 打开 Uart TX DMA Channel= /Mcl/Mcl/MclConfig/Uart_Dma_Tx
- 打开 Uart RX DMA Channel= /Mcl/Mcl/MclConfig/Uart_Dma_Rx

3.4 修改 Platform 模块

Uart_Example_S32G399A_M7->someId(…)->Platfrom(…)-> Platfrom->Interrupt Controller-> IntCtrlConfig->IntCtrlConfig：

->PlatformIsrConfig_83：LINFLEXD1_IRQn：

- Interrupt Enabled=unchecked//关掉 flexlin1 中断。
- Handler = undefined_handler

-> PlatformIsrConfig_8：DMA0_0_15_IRQn

- Interrupt Enabled=checked
- Priority (dynamic range)*=7
- Handler = Dma0_Ch0_Ch15_IrqHandler //参考源文件：mcal_ts_xxxx/src/Dma_Ip_Irq.c，及 S32G3 RM 附件：S32G3_DMAMUX_map.xlsx 的 DMA_CH_MUX_0 页：

5	Internal DMA TX request0	LinFlexD_1
6	Internal DMA RX request0	LinFlexD_1

所以是使用 DMA0 的低 Channel。

-> PlatformIsrConfig_10：DMA0_ERR0_IRQn

- Interrupt Enabled=checked
- Handler = Dma0_Error_IrqHandler

3.5 处理 Cache 相关问题

参考文档《RTD_UART_IM.pdf》的：5.7 节 Data Cache Restrictions 说明：

In the DMA transfer mode, DMA transfers may have cache coherency problems. To avoid possible coherency issues when **D-CACHE** is enabled, the user shall ensure that the buffers used as TCD source and destination are allocated in the **NON-CACHEABLE** area (by means of Uart_Memmap). Otherwise, the Uart driver has some dependencies. User must to put all variables, which were used for transmitter and receiver, in the **NON CACHEABLE** memory section in the RAM zone by the definition **UART_START_SEC_VAR_<INIT←-_POLICY>_<ALIGNMENT>_NO_CACHEABLE** and **UART_STOP_SEC_VAR_<INIT_PO←-LICY>_<ALIGNMENT>_NO_CACHEABLE**.

参考源代码：basenxp_tx_xxxx\include\Uart_MemMap.h：中定义了诸如：

#define UART_START_SEC_VAR_CLEARED_UNSPECIFIED_NO_CACHEABLE

的宏，用法如：Uart_ts_xxxx\src\LinFlexd_Uart_Ip.c

```
#define UART_START_SEC_VAR_CLEARED_UNSPECIFIED_NO_CACHEABLE
#include "Uart_MemMap.h"
/** @brief Array of UART driver runtime state structures */
Linflexd_Uart_Ip_StateStructureType Linflexd_Uart_Ip_apStateStructure[LINFLEXD_INSTANCE_COUNT];
#define UART_STOP_SEC_VAR_CLEARED_UNSPECIFIED_NO_CACHEABLE
```

所以我们修改源文件：
C:\NXP\SW32G_RTD_4.4_4.0.2\eclipse\plugins\Uart_TS_T40D11M40I2R0\examples\EBT\S32G3\Uart_Example_S32G399A_M7\src\main.c

```
#define UART_START_SEC_VAR_CLEARED_UNSPECIFIED_NO_CACHEABLE
#include "Uart_MemMap.h"
uint8 Rx_Buffer[MSG_LEN];
uint8 Tx_Buffer[MSG_LEN];
#define UART_STOP_SEC_VAR_CLEARED_UNSPECIFIED_NO_CACHEABLE
…
/ /   uint8 Rx_Buffer[MSG_LEN];
//    uint8 Tx_Buffer[MSG_LEN];
```

然后查看编译生成的 main.map 文件：

```
.mcal_bss_no_cacheable
            0x3450d3f8      0x64 out/main.o
            0x3450d3f8              Rx_Buffer
            0x3450d42a              Tx_Buffer
            0x3450d45c              . = ALIGN (0x4)
```

所以会把源和目的地址放在 nocacheable 的地址。

3.6 测试结果

与默认测试方法一样：

```
This example is an simple echo using UART
The board will greet you if you send 'Hello'
Now you can begin typing：  (此处输入 Hello\r)
Hello World
```